Programming Research Group

PROBABILISTIC DETREMINISM

Annabelle McIver, Carroll Morgan

PRG-TR-13-95

Oxford University Computing Laboratory
Wolfson Building, Parks Road, Oxford OX1 3QD

Probabilistic determinism

Annabelle McIver, Carroll Morgan

15 November 1995

Abstract

In [3] a probabilistic space PCSP of processes is constructed uniformly from the standard CSP failures-divergences model. Here we identify the deterministic (probabilistic) elements of PCSP as the maximal elements in the refinement ordering. We extend the standard characterisation of deterministic processes in terms of of properties of their finite approximations, and formulate some analogues of the standard syntactic rules for ensuring determinism.

1 Introduction

In the standard failures-divergences model of CSP [1], the maximal elements are of interest because they correspond to the programs whose behaviour is exactly specified, and thus what a programmer should implement. In the probabilistic model described in [3] the set of elements maximal in the refinement ordering not only contains all the standard (non-probabilistic) deterministic processes, but many that can make probabilistic choices. For example the process *Flip*,

$$Flip \quad := \quad (h \rightarrow Stop\,{}_{0.5}{\oplus}\; t \rightarrow Stop),$$

that models the single flip of an unbiased coin is maximal in PCSP. Although *Flip* may deliver different outcomes on different executions the outcomes are all subject to a single uniform probability distribution, and thus *Flip* cannot be further refined lest the refinement be biased.

It turns out that, just as in CSP, all non-determinism in PCSP is inherited from nondeterministic choice: essentially, maximal probabilistic processes can resolve only probabilistic choice, and not the demonic choice expressed by the operator $\sqcap$.

Returning for the moment to CSP, it is easy to see that a standard process is deterministic when it can make only deterministic decisions at every stage of its execution: that is, it is nondeterministic as a whole only if the nondeterminism is revealed at some finite point. That insight into the role of finiteness will guide us as we investigate probabilistic determinism — for we shall see that deterministic process in PCSP can only make either probabilistic choices or standard deterministic choices at each stage of its execution.

The outline of the paper is as follows. In Sec. 2 and Sec. 3 we look more closely at the general structure of PCSP. In Sec. 4 and Sec. 5 we characterise determinism in terms of observations over its finitary behaviour. Finally in Sec. 6 we prove analogues of the standard syntactic rules for ensuring determinism.

2 Finitary representations

Although PCSP is a complete partial order, it is alas not algebraic as CSP is. Nevertheless it does have a finitary sub-structure, comprising those PCSP processes that can be formed by finitely many uses of ${}_p\oplus$ applied to finite standard processes.

Definition 2.1 A *finitary* process in PCSP is one that can be written as a finite probabilistic combination of embedded finite standard (CSP) processes.

That set of finite standard processes is called the *support* of the finitary (probabilistic) process; the support together with the associated coefficients is called the *finitary representation.*

Usually (but not always) we assume that the processes in the support are given without repetition, and that the associated coefficients are non-zero. □

An example of such a process is

$$Flip \quad := \quad h \to STOP \;{}_p\!\oplus\; t \to STOP \ , \tag{1}$$

mentioned earlier. Its support is the two (standard) processes

$$h \to STOP \quad \text{and} \quad t \to STOP \ ,$$

each finite, with the associated probabilistic coefficients p and $\overline{p}$.

The importance of the finitary processes is shown by the following two lemmas, establishing that the representations are unique and that they form a 'basis' for PCSP.

Lemma 2.2 Finitary representations are unique (up to zero coefficients, repetition and reordering).

Proof: Let Φ, Γ be two vectors of reals in $(0,1]$ and let $\mathcal{F}, \mathcal{G}$ be two vectors of finite standard processes, without repetition, so that $(\Phi, \mathcal{F})$ and $(\Gamma, \mathcal{G})$ are finitary representations as in Def. 2.1.

For a contradiction, suppose that $(\Phi, \mathcal{F})$ and $(\Gamma, \mathcal{G})$ are distinct, yet satisfy

$$\Phi \cdot \overline{\mathcal{F}} \quad = \quad \Gamma \cdot \overline{\mathcal{G}} \ ,$$

where $\cdot$ denotes inner product of vectors and $\overline{\mathcal{F}}, \overline{\mathcal{G}}$ are the vectors of PCSP processes got by embedding each element of $\mathcal{F}, \mathcal{G}$.

By suitable subtractions from each side of the equation above form the vectors $\Phi', \mathcal{F}', \Gamma', \mathcal{G}'$ so that

$$\Phi' \cdot \overline{\mathcal{F}'} \quad = \quad \Gamma' \cdot \overline{\mathcal{G}'} \ , \tag{2}$$

and $\mathcal{F}', \mathcal{G}'$ have no elements in common. Since the representations are distinct (are not permutations of each other), we cannot have both $\mathcal{F}'$ and $\mathcal{G}'$ empty in (2); thus choose (ϕ, F), without loss of generality in $(\Phi', \mathcal{F}')$, so that F is maximal within $\mathcal{F}', \mathcal{G}'$.

Since CSP is algebraic we can find a finite H such that $H \sqsubseteq F$ but H is not refined by any process in $\mathcal{G}$. Applying each side of (2) to $H\!\uparrow$ shows that ϕ must be 0, a contradiction. □

Lemma 2.3 Every process in PCSP is the limit of a directed set of finitary PCSP processes.

Proof: It is shown in Theorem 5.2 of [2] that every PCSP process is the directed limit of finite linear combinations of embedded standard (not necessarily finite) processes; but since CSP is algebraic each of those can in turn be written as a directed limit of finitary processes.

The required directed set overall is found as the union of the constituent directed sets. □

Although Lem. 2.3 does suggest an 'algebraic character' for PCSP, in fact for arbitrary A we have that

$$A \quad = \quad (\sqcup p \mid 0 \leq p < 1 \cdot A\ {}_p\!\oplus\ \overline{CHAOS}) \ ,$$

and so the reason for PCSP's failure to be algebraic is thus that its only *ipo*-finite process is $\overline{CHAOS}$, and that in particular our finitary processes are not *ipo*-finite.

3 n-finitary processes and representations

We now sharpen our notion of finitary, just as we did earlier in the standard case. Recall that a finite standard process P is said more specifically to be *n-finite* whenever $P{\downarrow}n = P$ [3, Def. 5.2]; it is *finite* just when it is n-finite for some n.

We have the following analogue in PCSP.

Definition 3.1 A process A in PCSP is said to be *n-finitary* whenever

$$A\overline{{\downarrow}n} \quad = \quad A\ .$$

□

We must call such processes 'n-finitary' rather than 'n-finite', because they are not *ipo*-finite.

Definition 3.2 A finitary representation $(\Phi, \mathcal{F})$ is said to be *n-finitary* whenever every element of $\mathcal{F}$ is n-finite (in CSP). □

Although we court confusion by applying the adjective n-finitary both to processes (Def. 3.1) and to representations (Def. 3.2), our aim is now to show that the two uses are essentially the same (Thm. 3.5 below).

Our first implication is easy.

Lemma 3.3 If a process in PCSP has n-finitary representation then it is n-finitary itself.

Proof: We have for, A in PCSP,

$$\begin{array}{lll}
 & A\overline{{\downarrow}n} & \\
= & (\Phi \cdot \overline{\mathcal{F}})\overline{{\downarrow}n} & \text{representation} \\
= & \Phi \cdot (\overline{\mathcal{F}}\,\overline{{\downarrow}n}) & {\downarrow}n \text{ linear} \\
= & \Phi \cdot \overline{\mathcal{F}{\downarrow}n} & \\
= & \Phi \cdot \overline{\mathcal{F}} & \text{representation is } n\text{-finite} \\
= & A\ . &
\end{array}$$

□

The reverse implication is more involved, and depends on the characterisation of refinement between finitary processes given in [2].

Lemma 3.4 If a PCSP process is n-finitary then it has n-finitary representation.

Proof: From Lem. 2.3 we know that any A in PCSP is a limit of a directed set of finitary processes; applying $\overline{\downarrow n}$ to each element of that set gives a directed set of n-finitary representations, having limit $A\overline{\downarrow n}$ by continuity of $\overline{\downarrow n}$. Since A is n-finitary by assumption, that limit is simply A itself.

Thus we need only show that the set of PCSP processes with n-finitary representation is closed under directed limits.

Since there are only finitely many n-finite (standard) processes, we can for simplicity present them, all at once, as a vector $\mathcal{F}$, of length N say, and then identify the n-finitary representations with N-element vectors Φ over $[0,1]$, each Φ_i being the (possibly 0) coefficient of $\mathcal{F}_i$ in the representation. By Lem. 2.2 we have a bijection.

Assume the vector $\mathcal{F}$ is arranged so that $i \leq j$ whenever $\mathcal{F}_i \sqsubseteq \mathcal{F}_j$ (thus topologically sorted), and form the $N \times N$ matrix

$$\underline{\mathcal{F}}_{ij} \quad := \quad 1 \ \underline{\text{if}}\ (\mathcal{F}_i \sqsubseteq \mathcal{F}_j)\ \underline{\text{else}}\ 0 \ .$$

Clearly $\underline{\mathcal{F}}$ has ones along its main diagonal, and the ordering on $\mathcal{F}$ guarantees that only zeroes appear below; thus $\underline{\mathcal{F}}$ is invertible.

Now by Corollary 4.9 of [2] we have (using $\cdot$ also for matrix multiplication)

$$\Phi \cdot \overline{\mathcal{F}} \sqsubseteq \Phi' \cdot \overline{\mathcal{F}} \quad \text{iff} \quad \underline{\mathcal{F}} \cdot \Phi \leq \underline{\mathcal{F}} \cdot \Phi' \ ,$$

where the inequality on the right, between vectors of reals, is taken pointwise. Thus to find the directed limit, we need only multiply each element Φ by $\underline{\mathcal{F}}$, take the pointwise limit of those, and finally multiply that limit by $\underline{\mathcal{F}}^{-1}$; the result follows by continuity of the multiplications. □

Now we have the correspondence we seek, summed up in this theorem.

Theorem 3.5 A PCSP process is n-finitary iff it has n-finitary representation.

Proof: See Lem. 3.3 and Lem. 3.4. □

Finally, we find with Cor. 3.6 the probabilistic analogue we seek of the standard 'finite iff n-finite for some n'.

Corollary 3.6 A PCSP process is finitary iff it is n-finitary for some n.

Proof: If A is finitary, then it has finitary representation $(\Phi, \mathcal{F})$ say. Recall that $\mathcal{F}$ is a finite (in length) vector of finite processes; thus choose n large enough so that every element of $\mathcal{F}$ is n-finite. Then $(\Phi, \mathcal{F})$ is an n-finitary representation, and so A is n-finitary by Lem. 3.3.

If A is n-finitary, then by Lem. 3.4 it has n-finitary representation, and thus is trivially finitary. □

With the above tools, we can now show that (even probabilistic) nondeterministic, if present, must reveal itself at a finite stage.

4 Determinism

The key notion here is that of 'n-determinacy': we define a standard process to be n-determinate whenever it contains no nondeterminism (at least) for the first n steps. The characterisation generalises nicely to PCSP— for we have seen that every PCSP process is a limit of finitary processes, and so the strategy of examining the early behaviour for nondeterminism should work there as well.

We can regard a process as making a finite number of probabilistic choices between embedded finitary processes at each stage of its execution, and if each of those is n-determinate, at the n^{th} stage, then the limit will be deterministic. For example, the (recursive) process

$$Flips \quad := \quad h \to Flips \,{}_{0.5}\!\oplus\, t \to Flips$$

is maximal; and it is the limit of the chain which begins

$$\begin{array}{ll}
 & \overline{CHAOS} \\
\sqsubseteq & h \to \overline{CHAOS} \,{}_{0.5}\!\oplus\, t \to \overline{CHAOS} \\
 & \\
\sqsubseteq & h \to h \to \overline{CHAOS} \quad @\, 0.25 \\
 & h \to t \to \overline{CHAOS} \quad @\, 0.25 \\
 & t \to h \to \overline{CHAOS} \quad @\, 0.25 \\
 & t \to t \to \overline{CHAOS} \quad @\, 0.25 \;,
\end{array}$$

in which each process is n-determinate for the appropriate n (here 0, 1 and 2 respectively).

We begin, however, with determinism overall:

Definition 4.1 A process in PCSP is said to be *deterministic* whenever it is maximal in the refinement ordering. □

Now to introduce the notion of 'n steps' we consider the n-finitary processes: if an n-finitary process contains nondeterminism in its first n steps, then by removing that nondeterminism we can (properly) refine it while preserving its n-finitariness. Thus we make the following two definitions.

Definition 4.2 A PCSP process is said to be *n-maximal* if it is maximal among the n-finitary processes. □

Definition 4.3 A PCSP process A is said to be *n-determinate* if $A{\downarrow}n$ is n-maximal.[1] □

[1] An alternative definition of n-determinacy is as refinement of n-maximality: simply apply ${\downarrow}n$ to both sides.

As planned, an n-determinate process is one in which nondeterministic behaviour does not appear before the n^{th} step. (Note that Definitions 4.2 and 4.3 apply equally well to CSP, provided we replace 'finitary' by 'finite'.)

Now we concentrate on our desired characterisation of determinism in terms of n-determinacy. For that, however, we will need a detailed result from the standard CSP space; rather than complicate the structure here, however, we give its proof in the next section (Lem. 4.5 refers).

Lemma 4.4 If PCSP process is n-determinate for all n, then it is deterministic.
Proof: Suppose for the contrapositive that $A \sqsubset B$. Then for all n we have $A{\downarrow}n \sqsubseteq B{\downarrow}n$, with the inequality being strict for at least one n (else $A = B$). But if $A{\downarrow}n \sqsubset B{\downarrow}n$, then A is not n-determinate. $\square$

Lemma 4.5 If PCSP process is deterministic, then it is n-determinate for all n.
Proof: Suppose for the contrapositive that $A{\downarrow}n \sqsubset B{\downarrow}n$; then clearly $B{\downarrow}n \not\sqsubseteq A$. We shall form a chain $\mathcal{B}$ such that for all m both

$$A{\downarrow}m \sqsubseteq \mathcal{B}_m \quad \text{and} \quad B{\downarrow}n \sqsubseteq \mathcal{B}_m \ .$$

Then we will have $A \sqsubseteq \sqcup\mathcal{B}$ and $B{\downarrow}n \sqsubseteq \sqcup\mathcal{B}$, giving $A \sqsubset \sqcup\mathcal{B}$, as desired.

To form the chain $\mathcal{B}$ we use the consistency lemma, Lem. 5.3 of the next section, specialised as follows:

> For PCSP processes A, B and natural n, if $A{\downarrow}n \sqsubseteq B{\downarrow}n$ then there is a process C such that both $A{\downarrow}(n{+}1) \sqsubseteq C{\downarrow}(n{+}1)$ and $B{\downarrow}n \sqsubseteq C{\downarrow}(n{+}1)$.

With that principle the construction of $\mathcal{B}$ is simple. For $m \leq n$ set (trivially) $\mathcal{B}_m := B{\downarrow}n$. Since $A{\downarrow}n \sqsubseteq B{\downarrow}n$, we set $\mathcal{B}_{n+1} := C{\downarrow}(n{+}1)$ where C is given as above: thus both $A{\downarrow}(n{+}1) \sqsubseteq \mathcal{B}_{n+1}$ and $B{\downarrow}n \sqsubseteq \mathcal{B}_{n+1}$ as required.

For $\mathcal{B}_{n+2}$ we repeat the construction, starting from $A{\downarrow}(n{+}1)$ and $\mathcal{B}_{n+1}$, and carry on thus to form all of $\mathcal{B}$. $\square$

We conclude this section by putting the two results together.

Theorem 4.6 A PCSP process is deterministic iff for all n it is n-determinate.
Proof: Lem. 4.4 and Lem. 4.5 provide the necessary implications. $\square$

5 The consistency lemma

Here we provide the precise details of the construction necessary for Lem. 4.5. In the following lemmas we write $\mathrm{Fail}(P)$ for the failures of a standard process, $\mathrm{Tr}(P)$ for its traces and $\mathrm{Div}(P)$ for its divergences.

In the first lemma we state the facts we need concerning the extent to which $P{\downarrow}n$ and P agree for the first n steps.

Lemma 5.1 For process P, natural number n, trace s and refusal X, we have

1. $(s, X) \in \mathrm{Fail}(P{\downarrow}n)$ iff $(s, X) \in \mathrm{Fail}(P)$ whenever $\#s < n$; and
2. $s \in \mathrm{Tr}(P{\downarrow}n)$ iff $s \in \mathrm{Tr}(P)$ whenever $\#s \leq n$.

Proof: The result follows directly from the definition of $({\downarrow}n)$. □

Now the principal lemma for this section follows. Roughly speaking, it states (a slight generalisation of) the fact that if $P{\downarrow}n$ is not n-maximal, as witnessed by some refinement $P{\downarrow}n \sqsubset Q{\downarrow}n$, then for any greater m neither is $P{\downarrow}m$ m-maximal; moreover a witness to that latter can be found by 'mimicking' the refinement that led from $P{\downarrow}n$ to $Q{\downarrow}n$ (so that the m-maximal proper refinement of $P{\downarrow}m$ is a refinement of $Q{\downarrow}n$ also).

Lemma 5.2 Let P, Q be standard processes, and suppose for natural number n that

$$P{\downarrow}n \sqsubseteq Q{\downarrow}n \ .$$

Then $P{\downarrow}m$ and $Q{\downarrow}n$ are consistent, for any m.

Proof: If $m \leq n$ the consistency is obvious. For $m > n$, in fact we identify the least upper bound R of $P{\downarrow}m$ and $Q{\downarrow}n$, as follows:

$$\begin{array}{lcl} \mathrm{Fail}(R) & := & \mathrm{Fail}(P{\downarrow}m) \cap \mathrm{Fail}(Q{\downarrow}n) \\ \mathrm{Div}(R) & := & \mathrm{Div}(P{\downarrow}m) \cap \mathrm{Div}(Q{\downarrow}n) \ . \end{array}$$

In view of that definition, consistency of $P{\downarrow}m$ and $Q{\downarrow}n$ is established merely by showing that R is well formed — and in the case of a process constructed by intersection, the only well-formedness condition at risk is this:

> For all failures (s, X) in $\mathrm{Fail}(R)$, and events x,
> either $s \frown \langle x \rangle \in \mathrm{Tr}(R)$ or $(s, X \cup \{x\}) \in \mathrm{Fail}(R)$.

That is the condition, informally expressed, that a process must either accept or refuse an event: it cannot do neither.

We assume in the following that (s, X) is a failure of R, and hence is a failure of both $P{\downarrow}m$ and $Q{\downarrow}n$ as well. In the case that $\#s \geq n$, we have both $s^\frown\langle x\rangle \in \mathrm{Tr}(Q{\downarrow}n)$ and $(s, X \cup \{x\}) \in \mathrm{Fail}(Q{\downarrow}n)$ for all x and X, and the well-formedness of R then follows from the well-formedness of $P{\downarrow}m$.

If $\#s < n$ then we assume for a contradiction that R is not well-formed, and consider two subcases. First we have

$$
\begin{array}{lll}
 & (s, X \cup \{x\}) \notin \mathrm{Fail}(P{\downarrow}m) \wedge s^\frown\langle x\rangle \notin \mathrm{Tr}(Q{\downarrow}n) & \\
\text{hence} & & \#s \leq n < m,\ \text{Lem. 5.1} \\
 & (s, X \cup \{x\}) \notin \mathrm{Fail}(P{\downarrow}n)) \wedge s^\frown\langle x\rangle \notin \mathrm{Tr}(Q{\downarrow}n) & \\
\text{hence} & (s, X \cup \{x\}) \notin \mathrm{Fail}(Q{\downarrow}n) \wedge s^\frown\langle x\rangle \notin \mathrm{Tr}(Q{\downarrow}n) & P{\downarrow}n \sqsubseteq Q{\downarrow}n \\
\text{iff} & \text{false}\ . & \text{well-formedness of } Q{\downarrow}n
\end{array}
$$

Then for the other subcase we have

$$
\begin{array}{lll}
 & s^\frown\langle x\rangle \notin \mathrm{Tr}(P{\downarrow}m) \wedge (s, X \cup \{x\}) \notin \mathrm{Fail}(Q{\downarrow}n) & \\
\text{hence} & s^\frown\langle x\rangle \notin \mathrm{Tr}(P{\downarrow}m) \wedge s^\frown\langle x\rangle \in \mathrm{Tr}(Q{\downarrow}n) & \text{well-formedness of } Q{\downarrow}n \\
\text{hence} & s^\frown\langle x\rangle \notin \mathrm{Tr}(P{\downarrow}m) \wedge s^\frown\langle x\rangle \in \mathrm{Tr}(P{\downarrow}n) & P{\downarrow}n \sqsubseteq Q{\downarrow}n \\
\text{hence} & s^\frown\langle x\rangle \notin \mathrm{Tr}(P{\downarrow}m) \wedge s^\frown\langle x\rangle \in \mathrm{Tr}(P{\downarrow}m) & \#s < n < m,\ \text{Lem. 5.1} \\
\text{iff} & \text{false}\ . &
\end{array}
$$

□

Our final lemma — and the one we actually need — is merely the generalisation of Lem. 5.2 to the probabilistic case. It relies on our being able to 'pair off' the representations of two n-finitary processes, where one refines the other, so that the refinement is realised pointwise between the representing vectors.

Lemma 5.3 *Consistency lemma* Let A, B be probabilistic processes in PCSP, and suppose for natural number n that

$$A{\downarrow}n \sqsubseteq B{\downarrow}n\ .$$

Then $A{\downarrow}m$ and $B{\downarrow}n$ are consistent, for any m.

Proof: The case in which $m \leq n$ is trivial, as in Lem. 5.2. For case $m > n$ we use the *Splitting Lemma* [2, Thm. 4.10] to argue that we can find a vector Φ of reals in $[0, 1]$ and three vectors $\mathcal{A}, \mathcal{B}, \mathcal{A}'$ of finite standard processes so that

$$
\begin{array}{rcl}
A{\downarrow}n & = & \Phi \cdot \overline{\mathcal{A}} \\
B{\downarrow}n & = & \Phi \cdot \overline{\mathcal{B}} \\
A{\downarrow}m & = & \Phi \cdot \overline{\mathcal{A}'}\ ,
\end{array}
$$

and so that for all indexes i

$$\mathcal{A}_i \sqsubseteq \mathcal{B}_i \quad \text{and} \quad \mathcal{A}_i = \mathcal{A}'_i{\downarrow}n \ .$$

(Note that the process vectors may contain repeated elements, and that Φ may contain zeroes.)

The result then follows by applying Lem. 5.2 index-wise to the three vectors $\mathcal{A}, \mathcal{B}, \mathcal{A}'$ to form a vector $\mathcal{C}$, say: the desired process is then $\Phi \cdot \overline{\mathcal{C}}$. □

We should note at this point a further application of the technique used in the proof above. By analogy with Def. 4.2 and Def. 4.3, let a *standard* process P be said to be n-determinate if $P{\downarrow}n$ is maximal among the n-finite processes.

Lemma 5.4 An n-finitary PCSP process is (PCSP-) n-maximal iff all elements of its support are (CSP-) n-maximal.

Proof: If any process in the n-finitary representation of A is not n-maximal, then A itself can be properly refined by refining that component of the representation.

If A is not n-maximal, then $A \sqsubset B$ for some n-finitary B and as in the proof of Lem. 5.3 we can write

$$A \quad = \quad \Phi \cdot \overline{\mathcal{A}} \quad \sqsubset \quad \Phi \cdot \overline{\mathcal{B}} \quad = \quad B \ ,$$

with $\mathcal{A}_i \sqsubseteq \mathcal{B}_i$ as before, for each index i. Clearly that last inequality must be strict for some i. □

6 Probabilistic determinism

Our general results above have shown that when a probabilistic process A is written as the limit $(\sqcup n \cdot A{\downarrow}n)$ of what we might call its 'n-approximates', each $A{\downarrow}n$ is a finite probabilistic combination of (embedded) n-finite processes. Moreover, A is deterministic exactly when each of those n-finite processes are n-maximal.

That reduction 'of the infinite to the finite' gives us easy proofs of many of the useful facts about probabilistic determinism: that it is preserved by embedding and by probabilistic choice, and that 'deterministic recursive definitions' (in a sense made precise below) produce deterministic fixed points. First we treat embedding.

Lemma 6.1 A standard process P is (CSP-) deterministic iff its embedding $\overline{P}$ is (PCSP-) deterministic.

Proof:

	P deterministic	
iff	$(\forall n \cdot P{\downarrow}n$ is (CSP-) n-maximal)	standard CSP
iff	$(\forall n \cdot \overline{P}{\downarrow}n$ is (PCSP-) n-maximal)	see below
iff	$\overline{P}$ deterministic .	Thm. 4.6

For the deferred justification, note first that if $\overline{P}{\downarrow}n \sqsubset A{\downarrow}n$ for some probabilistic A, then we must have $P{\downarrow}n \sqsubset F$ for some n-finite standard F in the n-finitary representation of $A{\downarrow}n$.

The reverse implication is trivial. □

The following lemma, together with the one above, is enough to show that any probabilistic combination of (embedded) deterministic processes is deterministic.

Lemma 6.2 Probabilistic choice preserves determinism.
Proof: Suppose that A, B are deterministic PCSP processes. Then because

$$(A\,{}_p\!\oplus\, B){\downarrow}n \quad = \quad (A{\downarrow}n)\,{}_p\!\oplus\,(B{\downarrow}n)\ ,$$

the support of $(A\,{}_p\!\oplus\, B){\downarrow}n$ is contained in the union of the supports of $A{\downarrow}n$ and $B{\downarrow}n$.

Since A, B are deterministic, by Lem. 5.4 those latter supports contain only n-maximal elements; therefore the same holds for the former. □

In fact, Lem. 6.1 amd Lem. 6.2 are enough together to show that any non-recursive expression involving only deterministic standard processes, embedded determinism-preserving operators and probabilistic choice is deterministic: the probabilistic choices need only be be distributed to the outside of the expression.

For recursion, however, we need to consider to what extent determinism-preserving functions remain so when embedded. The notion of constructive function seems important for that: recall that a standard process-to-process function f is *constructive* iff for all n and P we have

$$f(P){\downarrow}(n{+}1) \quad = \quad f(P{\downarrow}n){\downarrow}(n{+}1)\ ,$$

with a similar definition applying in the probabilistic case. The following lemma shows why we are interested in constructiveness.

Lemma 6.3 If a process-to-process function $\mathcal{F}$: PCSP $\rightarrow$ PCSP is constructive and preserves determinism, then $\mu\mathcal{F}$ is deterministic.
Proof: We have first

$$
\begin{array}{lll}
 & (\mu\mathcal{F})\!\downarrow\! n & \\
= & (\sqcup m \cdot \mathcal{F}^m(\mathit{CHAOS}))\!\downarrow\! n & \text{Knaster-Tarski} \\
= & (\sqcup m \mid m \geq n \cdot \mathcal{F}^m(\mathit{CHAOS})\!\downarrow\! n) & \\
= & (\sqcup m \mid m \geq n \cdot \mathcal{F}^m(\mathit{STOP})\!\downarrow\! n)\ . & F \text{ is constructive; induction}
\end{array}
$$

Now each term $\mathcal{F}^m(\mathit{STOP})\!\downarrow\! n$ in the limit is n-maximal, because $\mathcal{F}^m$ preserves determinism and STOP is deterministic. Thus they are all equal, and the limit is just $\mathcal{F}^n(\mathit{STOP})\!\downarrow\! n$ — which is still n-maximal.

The result now follows from Lem. 4.4. $\square$

With Lem. 6.3 showing how to ensure that $\mu\mathcal{F}$ is deterministic, we now conclude by giving conditions under which an embedded function satisfies its conditions.

Lemma 6.4 Let $\mathcal{F}\colon \mathrm{CSP} \to \mathrm{CSP}$ be such that, for all n and $P\colon \mathrm{CSP}$, if P is n-maximal then $\mathcal{F}(P)$ is n-determinate. Then $\overline{\mathcal{F}}\colon \mathrm{PCSP} \to \mathrm{PCSP}$ preserves determinism.

Proof: Let $A\colon \mathrm{PCSP}$ be deterministic; then

$$
\begin{array}{lll}
 & \overline{\mathcal{F}}(A) & \\
= & (\sqcup n \cdot \overline{\mathcal{F}}(A\!\downarrow\! n)) & \\
\sqsupseteq & (\sqcup n \cdot \overline{\mathcal{F}}(A\!\downarrow\! n)\!\downarrow\! n) & \\
= & (\sqcup n \cdot \mathcal{B}_n)\ , & \text{each } \mathcal{B}_n\ n\text{-maximal, see below}
\end{array}
$$

which is a limit over n of n-maximal processes, and thus is itself deterministic. But if $\overline{\mathcal{F}}(A)$ refines a deterministic process, it is deterministic itself.

For the deferred justification, note that each $A\!\downarrow\! n$ is of the form $\Phi \cdot \overline{\mathcal{A}}$, with each element of $\mathcal{A}$ being n-maximal. Thus $\overline{\mathcal{F}}(A\!\downarrow\! n)$ is $\Phi \cdot \overline{\mathcal{F}(\mathcal{A})}$, and each element of $\mathcal{F}(\mathcal{A})$ is n-determinate by our assumption concerning $\mathcal{F}$. Thus $(\Phi \cdot \overline{\mathcal{F}(\mathcal{A})})\!\downarrow\! n$ is n-maximal as claimed. $\square$

Lemma 6.5 If $\mathcal{F}\colon \mathrm{CSP} \to \mathrm{CSP}$ is constructive

and determinism-preserving, then it satisfies the conditions of Lem. 6.4 (and hence $\overline{\mathcal{F}}$ is determinism-preserving also).

Proof: Suppose for a contradiction that P is n-maximal but $\mathcal{F}(P)$ is not n-determinate. Choose deterministic P' so that $P = P'\!\downarrow\! n$, and note that

$$
\begin{array}{lll}
 & \mathcal{F}(P')\!\downarrow\! n & \\
= & \mathcal{F}(P'\!\downarrow\! n)\!\downarrow\! n & \mathcal{F} \text{ constructive} \\
= & \mathcal{F}(P)\!\downarrow\! n\ , & \text{choice of } P'
\end{array}
$$

which by assumption is not n-maximal. But then $\mathcal{F}(P')$ is not deterministic, a contradiction. $\square$

Theorem 6.6 If $\mathcal{F}\colon \mathrm{CSP} \to \mathrm{CSP}$ is constructive and determinism-preserving, then so is $\overline{\mathcal{F}}$.

Proof: Lem. 6.5 and Lem. 6.4 shows that $\overline{\mathcal{F}}$ is determinism-preserving; in [3, Sec. 13] it is shown that $\overline{\mathcal{F}}$ is constructive if $\mathcal{F}$ is. $\square$

Lemmas 6.1, 6.2, 6.3 and Thm. 6.6 together show that probabilistic definitions give us deterministic processes just when we would expect them to: when, ignoring probabilistic choice, we would get determinism at the standard level.

References

[1] C.A.R. Hoare. *Communicating Sequential Processes.* Prentice Hall International, 1985.

[2] C. Jones. *Probabilistic Nondeterminism.* PhD thesis, Edinburgh University, 1990. Available as Technical Report ECS-LFCS-90-105.

[3] C.C. Morgan, A.K. McIver, K. Seidel, and J.W. Sanders. Refinement-oriented probability for CSP. Technical Report PRG-TR-12-94, Programming Research Group, August 1994.